CLAIMS

What is claimed is:

1. A method for managing distribution of messages for changing the state of shared data in a computer system having a main memory, a memory management system, a plurality of processors, each processor having an associated cache, and employing a directory-based cache coherency comprising the method of:

grouping the plurality of processors into a plurality of clusters;

tracking copies of shared data sent to processors in the clusters;

receiving an exclusive request from a processor requesting permission to modify a shared copy of the data;

generating invalidate messages requesting that other processors sharing the same data invalidate that data;

sending the invalidate messages only to clusters actually containing processors that have a shared copy of the data in the associated cache; and

broadcasting the invalidate message to each processor in the cluster.

2. The method of claim 1, wherein the invalidate message is sent to one master processor in a cluster, and further comprising:

the master processor distributing the invalidate message to one or more slave processors and waiting for an acknowledgement from said one or more processors;

if said one or more slave processors are configured to do so, distributing the invalidate message to one or more other slave processors, if any exist, and waiting for an acknowledgement from said other slave processors;

a slave processor which does not distribute the invalidate message to any other processor replying with an acknowledgement to the processor from which the invalidate message was received; and

upon receiving acknowledgements from all processors to which the invalidate messages were sent, a slave processor replying with an acknowledgement to the processor from which the invalidate message was received;

wherein upon receiving an invalidate message, the processor invalidating a local copy of the shared data, if it exists, and wherein upon receiving acknowledgements from all slave processors to which the invalidate messages were sent, the master processor sending an invalidate acknowledgment message to the processor that originally requested the exclusive rights to the shared data.

3. The method of claim 2, wherein:

the slave processors to which the master processor distributes the invalidate message are determined by data registers associated with the master processor; and

any other slave processors to which the slave processors distribute the invalidate message are determined by data registers associated with each slave processor;

wherein data registers exist and may be unique for each processor entry port.

4. The method of claim 2, wherein:

tracking of the shared copies of the data sent to the clusters is performed by setting a bit in a data register with at least as may bit positions as there are clusters;

wherein each cluster is associated with one bit position in the data register.

5. The method of claim 4, wherein sending the invalidate messages only to one master processor in a cluster actually containing processors that have a shared copy of the data in the associated cache further comprises the steps of:

selecting only the bit positions containing a set bit;

cross referencing the bit positions with cluster numbers;

cross referencing cluster numbers with an actual processor identification; and

delivering the invalidate message to the processor associated with the processor identification.

6. The method of claim 1, further comprising:

distributing the main memory among and coupled to each of the plurality of processors and each processor comprising a directory controller for the main memory coupled to that processor;

the directory controller managing the main memory location for the shared data and tracking the copies of shared data sent to processors in the clusters;

the processor requesting exclusive ownership of the shared data delivering the request to the directory controller; and

the directory controller sending the invalidate messages to master processors in clusters actually containing processors that have a shared copy of the data.

7. The method of claim 1, further comprising:

upon receiving a request from a processor requesting permission to modify a shared copy of the data, sending a response to the requesting processor indicating the number of additional shared copies of the data;

changing the state of the shared data by the requesting processor from shared to exclusive; and

waiting to modify the exclusive data until acknowledgements arrive from the clusters actually containing processors that have a shared copy of the data in the associated cache.

8. The method of claim 7, wherein:

when the processor requesting exclusive ownership of the shared data, the directory controller, and shared copies of the data exist within the same cluster, the directory node assumes the position of master node and broadcasts the invalidate message to all the processors in the cluster.

9. A multiprocessor system, comprising:

a main memory configured to store data;

a plurality of processors, each processor coupled to at least one memory cache;

a memory directory controller employing directory-based cache coherence;

at least one input/output device coupled to at least one processor;

a share mask comprising a data register for tracking shared copies of data blocks that are distributed from the main memory to one or more cache locations; and

a PID-SHIFT register which stores configuration settings to determine which one of several shared data invalidation schemes shall be implemented;

wherein when the PID-SHIFT register contains a value of zero, the data bits in the share mask data register correspond to one of the plurality of processors and wherein when the PID-SHIFT register contains a nonzero value, the data bits in the share mask data register correspond to a cluster of processors, each cluster comprising more than one of the plurality of processor.

10. The system of claim 9 wherein:

if the value in the PID-SHIFT register is zero, the directory controller sets the bit in the share mask corresponding to the processor to which a shared copy of a data block is distributed; and

wherein if the value in the PID-SHIFT register is nonzero, the directory controller sets the bit in the share mask corresponding to the cluster containing a processor to which a shared copy of a data block is distributed.

11. The system of claim 10 wherein:

the nonzero value in the PID-SHIFT register determines the number of processors in each cluster.

12. The system of claim 9 wherein:

when more than one shared copy of a data block exists outside of the main memory; and

wherein in response to a request from a requesting processor for exclusive write access to one of the shared copies of the data block; and

wherein when the value in the PID-SHIFT register is zero, the directory controller transmits an invalidate message only to those processors whose corresponding bits in the share mask are set, except the requesting processor; and

wherein when the value in the PID-SHIFT register is nonzero, the directory controller transmits an invalidate message only to those clusters whose corresponding bits in the share mask are set.

13. The system of claim 12 wherein the cluster further comprises:

a master processor to which the invalidate message directed toward the cluster are delivered; and

one or more slave processors, each of which receive an invalidate message that is generated by the master processor.

14. The system of claim 13 further comprising:

a processor router table that includes cross reference information which correlates master processor identification with cluster numbers.

15. The system of claim 13 further comprising:

configuration registers associated with each port of a processor in a cluster which determine the path by which the invalidate message is broadcast within a cluster.

16. A multiprocessor system, comprising:

a memory;

multiple computer processor nodes, each with an associated memory cache; and

a memory controller employing a directory-based cache coherency employing shared memory invalidation method, wherein:

the nodes are grouped into clusters;

the memory controller distributes memory blocks from the memory to the various cache locations at the request of the associated nodes;

upon receiving a request for exclusive ownership of one of the shared memory blocks, the memory controller distributes invalidate messages via direct point to point transmission to only those clusters containing nodes that share a block of data in the associated cache; and

wherein when the invalidate message is received by a cluster, an invalidate message is broadcast to all nodes in the cluster.

17. The system of claim 16 further comprising:

a share mask data register with as many bit locations as there are clusters;

a router lookup table with cross reference information correlating bit locations in the share mask to one master nodes in each cluster;

wherein the memory controller determines to which cluster to send the invalidate message according to bits set in the share mask and sends the invalidate message to the router which then forwards the invalidate message to the node whose identification corresponds to the cluster number as indicated in the router table.

18. The system of claim 17 each node further comprising:

router control and status registers for each input port of the node which configure the node's broadcast forwarding scheme wherein the forwarding scheme determines to which, if any, nodes the node shall forward a broadcast invalidate message when a broadcast invalidate message is received at a given port.

19. The system of claim 18 wherein:

the router control and status registers are comprised of bit locations corresponding to each output port of the node; and

wherein if a bit location contains a set bit, the invalidate message is forwarded to the output port corresponding to that bit location; and

wherein if a bit location does not contain a set bit, the invalidate message is not forwarded to the output port corresponding to that bit location.

20. The system of claim 19 wherein the processors in a cluster invalidate shared data, if it exists, and generate and forward acknowledgments in reverse direction but along the same path followed by the invalidate messages.